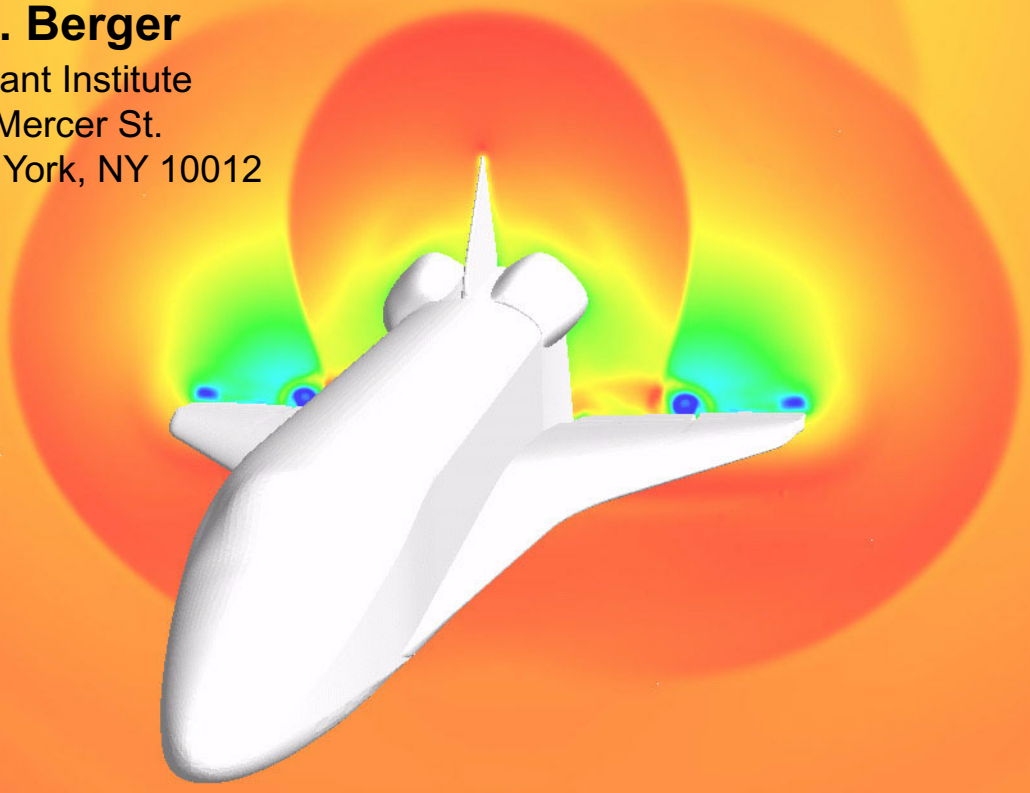
AIAA 2002-0863

Multilevel Error Estimation and Adaptive *h*-Refinement for Cartesian Meshes with Embedded Boundaries

M. J. Aftosmis
Mail Stop T27B
NASA Ames Research Center
Moffett Field, CA 94035

M.J. Berger
Courant Institute
251 Mercer St.
New York, NY 10012



40th AIAA Aerospace Sciences Meeting and Exhibit
14-17 January 2002 / Reno NV

Multilevel Error Estimation and Adaptive *h*-Refinement for Cartesian Meshes with Embedded Boundaries

M. J. Aftosmis[†]
Mail Stop T-27B
NASA Ames Research Center
Moffett Field, CA 94035
aftosmis@nas.nasa.gov

M. J. Berger[‡]
Courant Institute
251 Mercer St.
New York, NY 10012
berger@cims.nyu.edu

This paper presents the development of a mesh adaptation module for a multilevel Cartesian solver. While the module allows mesh refinement to be driven by a variety of different refinement parameters, a central feature in its design is the incorporation of a multilevel error estimator based upon direct estimates of the local truncation error using τ-extrapolation. This error indicator exploits the fact that in regions of uniform Cartesian mesh, the spatial operator is exactly the same on the fine and coarse grids, and local truncation error estimates can be constructed by evaluating the residual on the coarse grid of the restricted solution from the fine grid. A new strategy for adaptive h-refinement is also developed to prevent errors in smooth regions of the flow from being masked by shocks and other discontinuous features. For certain classes of error histograms, this strategy is optimal for achieving equidistribution of the refinement parameters on hierarchical meshes, and therefore ensures grid converged solutions will be achieved for appropriately chosen refinement parameters. The robustness and accuracy of the adaptation module is demonstrated using both simple model problems and complex three dimensional examples using meshes with from 10^6 to 10^7 cells.

1 Introduction

WHILE error estimation using local gradient recovery techniques has long been popular in structural mechanics and other disciplines governed by elliptic systems,[1] such rigorous error estimates are generally not available in fluid mechanics where the governing equations contain non-self-adjoint operators. As a consequence, error estimation and adaptation for fluid mechanics has evolved over a different path. The simplest methods in the literature are gradient and undivided difference-based feature detectors.[2][4] While these methods have been extremely successful for various classes of problems,[2]-[7] their lack of formalism makes them difficult to apply blindly to problems far from established experience. Indicators based upon estimates of interpolation error[7][8][9] lend some of the missing formalism. However, since these essentially compute the local curvature of a representative variable or combination of variables, they are not clearly superior (or different from) straight feature detection.[5] The multilevel error estimators from the literature on Adaptive Mesh Refinement (AMR) were among the first Richardson extrapolation-like error estimators[10][11][12] but have been only narrowly used outside the AMR community. More recently, there has been interest in multilevel error indicators which use the residual of a higher order interpolation of the existing solution to estimate the local error.[13][14][15] This approach is attractive since it can be combined with solution of an adjoint problem to produce a mesh adaptation strategy that seeks to minimize error in a specific integral quantity of engineering interest.[13]-[16] The hope of these techniques is that, for the cost of the adjoint solution and Jacobian storage, one can produce a mesh that reduces only those errors that are important to the problem at hand, rather than simply equidistributing the error.

While the research on adjoint-based mesh optimization is very exciting, two drawbacks make it currently unattractive for a general purpose analysis code. First, if the Jacobians are not already present in the solver, it is an expensive proposition to form them solely to drive the adaptation. Secondly and more fundamentally, many problems of interest have competing requirements that cannot always be encapsulated into a single output functional. For example, a user may wish to optimize a single simulation for lift, drag and pitching moment. While multipoint optimization is currently an area of research interest, results are not yet in-hand.

In this paper, we develop a new multilevel adaptation module for use on adapted Cartesian meshes with embedded boundaries. Figure 1 shows a sample hierarchy of such meshes used by a new, parallel multigrid solver.[17] The fact that these coarse meshes can be automatically generated makes multilevel error estimation a viable alternative, even for grids around very complex geometries. Seeking to take advantage of this fact, we explore Richardson extrapolation-like error estimators which exploit the hierarchical nature of adapted Cartesian grids. Even without the sensitivity information provided by adjoint-based approaches, the module will still offer huge savings over *a priori* mesh enrichment. The estimators we present are light-weight and both fast to implement and execute. They also leverage much of the machinery already built into many multigrid solvers.

Since Cartesian grid refinement is restricted to cell subdivision (*h*-refinement), the approach is intrinsically discrete. There exists a necessary reliance upon a refinement threshold which selects cells destined for subdivision. Our work examines the topic of threshold selection in detail. We develop a methodology for robustly setting a threshold which also ensures consistency of the adaptive process. The approach both controls the level of error in the domain and equidistributes the remaining error as fast as possible within the restrictions of hierarchical (nested) refinement. Given a robust strategy for setting the adaptation threshold, adaptive cases

[†] Research Scientist, Senior Member AIAA

[‡] Professor, Courant Institute, Member AIAA

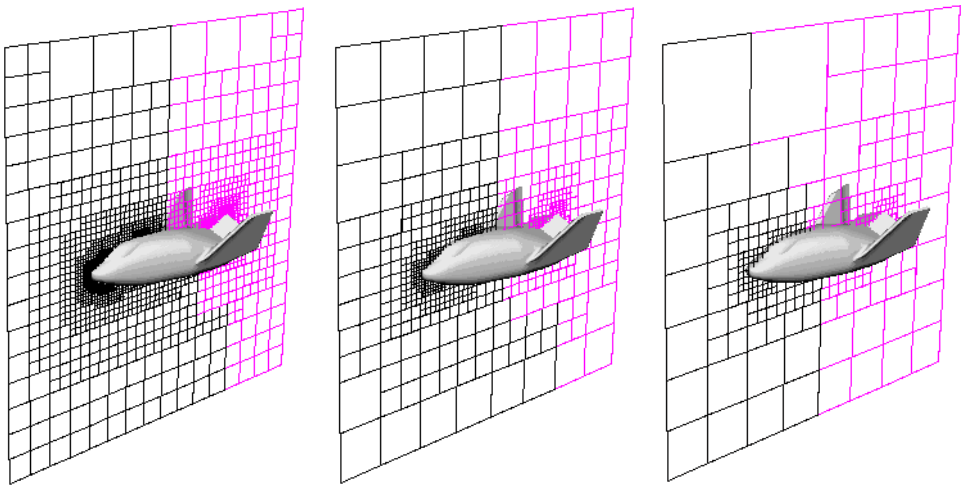


Figure 1: Hierarchical sequence of partitioned meshes around lifting body configuration used by parallel flow solver of ref. [17].

may be fully automated. This reduces the sensitivity of the results to user experience/intervention which has long been a drawback of adaptive methods.

2 Multilevel Error Estimation

Multilevel Richardson-type error estimators like those in [12] are attractive because of their low cost and firm theoretical underpinnings in smooth solutions. In addition to developing refinement parameters based on these estimates, we also develop first and second-difference based feature detectors which can serve useful roles as refinement parameters in practical examples.

2.1 Local Truncation Error Measurement

Consider the 1-D wave equation $u_t + Au_x = 0$ discretized with forward Euler in time and some numerical spatial difference scheme using a timestep k, and a cell dimension h,

$$\frac{1}{k}(U_j^{n+1} - U_j^n) + \frac{1}{h}(F_{R,j} - F_{L,j}) = 0\,. \qquad (1)$$

U_j^n is the discrete approximation to the continuous solution, $u(x, t)$, in cell j at time n, while $F_{L,j}$ and $F_{R,j}$ are the numerical fluxes through the left and right boundaries of j and contain the details of the spatial operator. The difference of the numerical fluxes is the residual and the discrete equation can be written more compactly as

$$\frac{1}{k}(U_j^{n+1} - U_j^n) + \frac{R(U_j^n)}{h} = 0\,, \qquad (2)$$

where the discrete residual operator $R(\bullet)$ now contains all details of the numerical flux balance for cell j.

The *local truncation error*, *LTE*, measures how well the discrete equation models the actual PDE throughout the domain. It is defined by replacing the approximate solution U_j^n with the exact solution $u(x, t)$ in equation (2). Since U_j^n exactly satisfies the difference equation, the exact solution will not, and the difference will be the local truncation error, $LTE_{h,k}(x,t)$.

$$LTE_{h,k}(x, t) = \frac{1}{k}[u(x, t+k) - u(x, t)] + \frac{R(u(x, t))}{h} \qquad (3)$$

If the problem reaches a steady state, then $u(x, t+k) = u(x, t)$ and we can drop the dependence on time.

$$LTE_h(x) = \frac{R(u(x))}{h} \qquad (4)$$

Equation 4 is instructive since it relates the local truncation error to the discrete residual operator $R(\bullet)$ and the cell dimension h. Furthermore is clear from the derivation that in multiple dimensions $LTE(\vec{x}) \approx {Residual}/{h^d}$ where d is the number of spatial dimensions of the domain.

2.2 Local Truncation Error on Uniform Cartesian Meshes

Equation 4 is useful since it permits direct measurement of the local truncation error of a particular numerical method on an actual computational grid for any problem that has an exact solution. One such example is the supersonic vortex model problem used in reference 18. Figure 2 outlines the problem and shows 4 sample telescoping meshes. The actual meshes used had from 130 to 7800 control volumes. After initialization with the exact solution, the local truncation error was computed within each cell by dividing the residual by the cell volume (following §2.1 and eq. (4)) using the Euler solver in reference 17 without limiters.

The L_1 norm of the *LTE* in density was computed for each mesh and is displayed in figure 3 as a function of the number of cells in each mesh. Performing a regression analysis on the Cartesian mesh data reveals a slope of 2.11. For smooth solutions, one can show that the global error is also second-order[19].

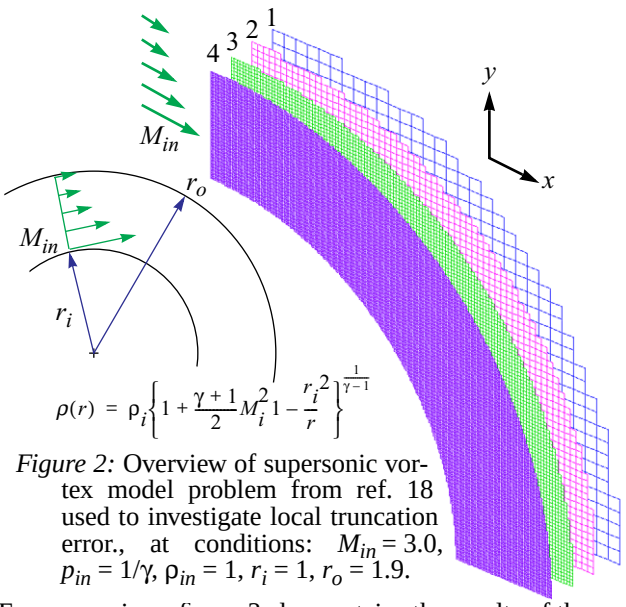


Figure 2: Overview of supersonic vortex model problem from ref. 18 used to investigate local truncation error., at conditions: $M_{in} = 3.0$, $p_{in} = 1/\gamma$, $\rho_{in} = 1$, $r_i = 1$, $r_o = 1.9$.

For comparison, figure 3 also contains the results of the same experiment performed with three types of body-fitted grids: (1) regular, nearly unit aspect ratio quads, (2) right triangles made by subdividing the quads, and (3) a "quality" triangulation[20] in which no angle was less than 29°. All methods used the same numerical scheme. While the details of this experiment are contained in the Appendix, figure 3 displays the results. The asymptotic slopes of all four mesh types are shown in the figure. We note that the slopes of all but the "quality" mesh are similar but that the level of error on the Cartesian grids is consistently 6 to 10 times lower than the other grids over the full range of problem sizes. While these results may be too narrow to generalize, this figure empahsizes the fact that since *LTE* measurement is bascally a flux balance, even minute irregularities in the mesh can produce

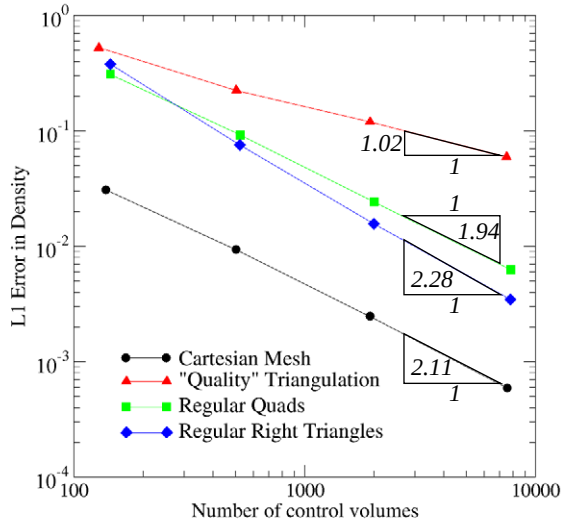


Figure 3: L_1 norm of *LTE* in density for supersonic vortex model problem on sequence of 4 meshes. The asymptotic slope of each curve is labeled. Further details of this experiment are presented in Appendix A.

subsantially higher local errors since we depend on the subtraction of two large numbers to produce a near zero result.

2.3 Estimation of *LTE* on Nested Multilevel Meshes

Since the *LTE* can be computed for each cell in the domain, it provides an excellent basis for constructing a refinement parameter within each cell. Unfortunately, since few problems of practical interest have an exact solution, we must develop a method of estimating the *LTE* within each cell.

The fact that we could reasonably draw straight lines through the data in figure 3 established the local and global order of the method. We can express this more formally by recalling that for a discrete method with local order p and a smooth solution, there exists a constant C_{lte} such that

$$\left\| LTE_h(\vec{x}) \right\| \le C_{lte} h^p . \tag{5}$$

Section *2.2* also established that for the solver in reference 17, p was around 2 when $\|\bullet\|$ is the 1-norm. Furthermore, since the RHS of equation 5 vanishes for small h, the method is obviously also consistent. Consistency implies that when h is small, the discrete solution $U_{j,h}$ in cell j of a mesh will be a good approximation to the exact solution $u(\vec{x})$, and this statement is the basis of our multilevel error estimation procedure.

Assume that the numerical method has converged on a fine grid with mesh spacing h so that the residual of the discrete solution will be zero for every cell j in the domain.

$$R_h(U_j) = 0 \tag{6}$$

Since the method is consistent, the current discrete solution is assumed to be close to the exact solution, $U_{j,h} \cong u(\vec{x})$. An estimate of the local truncation error on a coarse grid with mesh spacing H can then be written by substituting the discrete solution on the fine grid for the exact solution in equation 4.

$$LTE_H(\vec{x}) = \frac{R_H(I_h^H U_{j,h})}{H} \tag{7}$$

Following equation 7 we obtain an estimate of the *LTE* on the coarse grid by first restricting the discrete solution on the fine grid $U_{j,h}$ to a coarser grid using the interpolation operator I_h^H, and then evaluating the discrete residual of the restricted solution on the coarse grid. Since $U_{j,h}$ satisfied the numerical scheme on the fine grid, the restricted solution $I_h^H U_{j,h}$ will not, in general, produce exactly zero residual on the coarse grid. Thus, in the same way that the exact solution provided a means of measuring LTE_h in §2.1, the discrete solution on the fine grid provides a method of estimating the error on the coarse grid. Adaptively refined Cartesian meshes nest exactly. Therefore, the residual operator R is *exactly* the same on coarse and fine meshes. This permits us to apply equation 5 on a cell-by-cell basis. The coarse grid estimates are then used to trigger cell refinement on fine grid. This method of error estimation is known as τ-*extrapolation* within the multigrid community,[21] and the perfectly nesting residual opera-

tors that occur on Cartesian grids makes its use extremely attractive here. Related work was done in [12] using a different estimation technique.

In the present work, the restriction operator I_h^H is a volume weighted average, which is the same as that used by the multigrid smoother used for convergence acceleration. By using the prolongation operator from the multigrid scheme as well, we are able to construct refinement parameters on the fine mesh for the cost of one restriction, one coarse mesh residual evaluation and one prolongation. Since these operations use machinery which already exists in the multigrid solver, they may be implemented with little effort.

2.4 Mesh Irregularities

The preceding section examined error estimation on hierarchies of uniform Cartesian meshes. Adaptively refined Cartesian meshes, however, introduce some real-world complications which need discussion. As mentioned in the final paragraphs of §2.3, the multilevel error estimation procedure relies upon use of the same spatial operator R on both the fine and coarse meshes. While R_H and R_h are clearly the same in uniform regions of the grid, at adaptation boundaries and cut-cells the situation merits closer examination.

Figure 4 depicts a sample fine grid (a) and the corresponding coarse cells (b) near a simple refinement boundary. The adaptation boundary introduces irregularities into the connectivity graph which locally changes the residual operator. Thus, on the fine mesh (fig. 4a) all the cells which are immediately adjacent to the adaptation boundary have irregular stencils. The situation is the same on the coarse mesh (Fig. 4b) and since the residual operators on the two meshes must be identical for equation (5) to hold, the multilevel error estimator from §2.3 can't be accurately constructed for the fine cells shown shaded in figure 4a. Since this difficulty is associated with mesh irregularity, it appears in the residual operator of cut-cells as well.

While the formal basis for local truncation error estimation makes it preferable to straight feature detection, there are several drawbacks that make them difficult to use - even for Cartesian multilevel meshes. First, for a reliable error estimate, the same stencil needs to be applied to both the coarse and fine grids. Thus, at mesh irregularities such as grid interfaces and cutcells, the procedure described above does not produce a reliable estimate. Colella *et al.*[22] present a nice treatment of this problem in which the estimate is multiplied by the number of cells of that type. In essence, this scales a cell's contribution to the error by the number of affected cells. While the local truncation error at an interface is larger than at a uniform cell of the finer or even the coarser level, related work on convergence theory for non-uniform grids suggests that the estimate is still overly pessimistic.[23][24] (These results point out that its possible for $O(1)$ *LTE* to still preserve $O(h)$ global accuracy.) At present, we simply ignore error estimates at interfaces and cut cells, and rely on a buffering procedure to supply error values using piecewise constant extrapolation of the error estimates from neighboring cells.

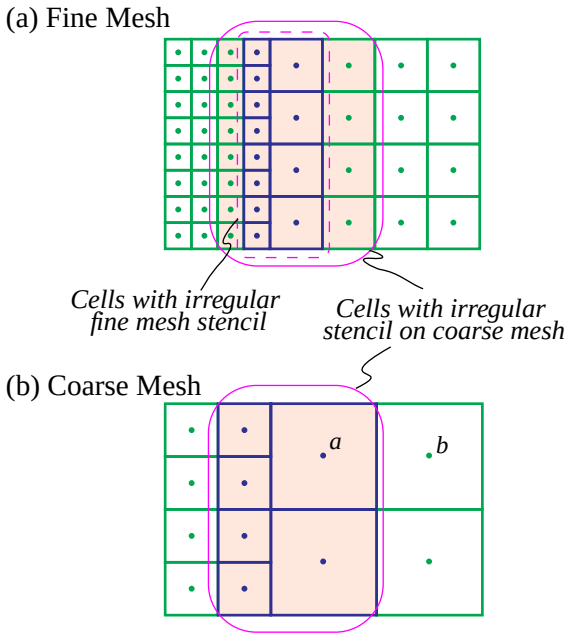


Figure 4: Illustration of difference stencils against adaptation boundary between cells of different levels. Cells in the shaded region of the fine mesh restrict their solution into the corresponding coarse mesh shown. Since the residual operator on this coarse mesh is different from that in the corresponding cells on the fine mesh, eq.5 is not valid.

Inspection of figure 4 demonstrates that this irregularity affects a larger portion of the mesh than one may initially suspect since the cell next to an interface has its gradient calculation affected. This cell in turn contaminates the stencil of a cell two away from the interface. As was demonstrated in figure 3, even a minor irregularity like this can lead to dramatically higher *LTE*. Additionally, since it is the coarse grid which provides the estimates, the pollution can extend as far as 4 cells from the interface on the fine grid, so that the fine grid cells have no reliable estimate of their own (*cf.* fig. 4). To produce grids with a stencil of 5 regular cells in each direction, large regions of uniformly refined cells need to be generated in the mesh. Although τ-estimates have been used with great success in block structured AMR methods, they are somewhat more difficult to implement on unstructured multilevel Cartesian. Asymptotic arguments contend that irregular cells are confined to only $O(N^2)$ cells in a 3-D mesh with $O(N^3)$ cells. In practice however, even reasonably fine meshes can have as many as 30% of the cells contaminated by irregularity.[1] When the neighbors of cells directly affected are also counted, *LTE* estimates on over half the mesh may have some contamination.

A second problem with *LTE* estimates is the need for a somewhat reliable solution to have already been computed. The

1. The situation is worse on coarse meshes where the difference between N^3 and N^2 is not as great.

approximate solution should be in the asymptotic region for a valid error estimate. This observation leads to a somewhat different strategy than that usually followed when using feature detection. Feature detectors generally perform well even when starting from a very coarse grid, making possible many cycles of adaptive refinement. For the *LTE* based refinement, our strategy is to start with a fairly refined initial grid, and only use two or three additional cycles of *LTE* driven adaptive refinement. In the examples here the initial grid uses geometry based refinement only, but initial grids generated from feature-detection based refinement could also be used.

2.5 Prespecified Adaptation Regions

Both the *LTE* estimates and feature detection approaches suffer from the difficulty of localizing them to regions of interest. For example, errors in the wake behind a wing persist for quite a distance downstream. However, if a user is interested in the pressure distribution on the wing, research suggests that the wake region doesn't need much mesh refinement. In the present work we simply allow the user to define a prespecified (Cartesian) region in which adaptation is permitted. While not particularly elegant, this approach is adequate in practice.

2.6 Feature Detection

Feature detection attempts to use the current discrete solution to determine where the mesh needs enhancement. Common schemes use first or second-order undivided differences and gradient information of various flow quantities. They attempt to "smooth out" computational space in hopes of producing a uniform error distribution.

Since the link between flow features and truncation error is not as formal as the Richardson-like *LTE* estimates discussed earlier, approaches in the literature vary significantly and everything from gradients and second derivatives to unscaled and scaled differences have been used.[2][3][4][5][26]

In 1991 Warren *et al.*[26] showed that since gradients and second derivatives stay approximately constant with mesh refinement, they make poor refinement parameters. A cell with a high gradient will continue to have a high gradient even after subdivision. From the standpoint of grid convergence, we note that if the refinement parameter is to act as a substitute for the real *LTE* it should have the same asymptotic behavior as the *LTE* in smooth regions of the flow. For a p^{th}-order scheme, this means that halving the mesh should reduce the error by 2^p.

The *LTE* of a second-order scheme reduces by a factor of 4 with 2:1 refinement. For a first-difference based quantity to mimic this behavior it must be scaled by the local mesh size, *h*. In one dimension, this leads to a first difference based refinement parameter of the form:

$$r_j = h_j \frac{\Delta\phi_j}{\phi_j} = h_j^2 \frac{\nabla\phi_j}{\phi_j} \tag{8}$$

which is a normalized version of the first difference parameter advocated by Warren *et al.*[26]

Similarly, a second difference based parameter should remain undivided to give the same behavior.[1]

$$r_j = \frac{\delta^2\phi_j}{\phi_j} = \frac{\phi_{j+1/2} - 2\phi_j + \phi_{j-1/2}}{\phi_j} = h_j^2 \frac{\nabla^2\phi_j}{\phi_j} \tag{9}$$

The 1-D refinement parameters in eqs. (8) and (9) can be extended using a finite volume approach. This produces a vector refinement parameter, with components for each of the cell's dimensions. The first difference based refinement parameter in the k^{th} direction of cell *j* is:

$$r_{j,k} = h_{j,k}^2 \frac{(\nabla\phi_j \bullet \hat{k})}{\phi_j} \tag{10}$$

where $\hat{k}$ is the unit vector in the k^{th} direction.

These vector refinement parameters can be used to drive both isotropic and anisotropic cell subdivision as discussed in reference [4], and a similar approach may be used for the second difference based parameters.

Popular choices of ϕ are the density, velocity magnitude or sometimes the local static pressure. In addition, combinations of these scalars are also possible. The investigations in section 4 examine the use of both density and velocity magnitude for detection.

3 An Optimal Strategy for *h*-Refinement

The adaptation strategy seeks to refine the mesh using the *LTE* estimates or other refinement parameters from §2 to improve the solution. The algorithm takes the refinement parameters from section 2 as input and returns a boolean refinement tag for each cell in the fine mesh. Algorithm **A** outlines the adaptation procedure.

Algorithm A: Adaptation Strategy

Input: Vector of normalized refinement parameters for each cell on fine mesh, $\hat{r}$

Output: Vector of cells tagged for *h*-refinement $\bar{\tau}$.

A.1 *Tag:* Apply the adaptation criteria $a(\bullet)$ to the normalized refinement parameter to produce a vector of tagged cells. $\bar{\tau} = a(\hat{r})$

A.2 *Rules:* Modify the set of tagged cells to ensure the validity and smoothness of the output mesh.

A.2.1 *Buffer*: Add buffer layers of tagged cells. $\bar{\tau} \leftarrow r_b(\bar{\tau})$

A.2.2 *Smoothness*: Filter for island/void suppression. $\bar{\tau} \leftarrow r_f(\bar{\tau})$

A.2.3 *Validity:* Ensure adaptation boundaries do not exceed 2:1. $\bar{\tau} \leftarrow r_l(\bar{\tau})$

A.3 *Output* final vector of tagged cells $\bar{\tau}$ for subdivision.

1. Eq. (9) differs from that presented in ref. [26] (eq. 15) since their parameter is re-scaled by the local mesh dimension and will therefore vanish *faster* than the *LTE* as the mesh size is decreased.

The final set of tagged cells output in **A.3** is then *h*-refined and the solution vector is initialized on the new fine mesh using the prolongation operator from the multigrid scheme. Taking this new fine mesh as input, the automatic coarse mesh generator in [17] prepares the multigrid mesh hierarchy, and the solver restarts.

In the following sub-sections we present a new strategy for reducing the refinement parameters to some predetermined level. The method is optimal since accomplishes this task as fast as possible. In addition the method ensures that the permissible level of the refinement parameter chosen at the outset will not vary as the mesh and solution evolve.

3.1 Equidistribution of Refinement Parameters

Equidistribution aims at producing a mesh which contains the same level of refinement parameter in each cell. Since the refinement parameters are stand-ins for the local truncation error, this goal spreads the remaining discretization error out evenly over the domain. As this level is reduced, the method is guaranteed to converge to the correct solution. This principle guards against over-resolving some features of the flow while overlooking others.

In practice, equidistribution is somewhat over-conservative most of the time. It assumes that all errors are equally important to the simulation, and this is certainly not the case most of the time. However, without additional guidance about what is important for a particular simulation, equidistribution simply ensures that everything in the simulation is equally correct. In addition, if a method can control the *LTE* distribution to achieve equidistribution, then it can control the error to achieve a different goal. Its easy to conceive of inverse-distance weightings or error weightings that take the local characteristics into account in order to identify those errors that have the strongest impact on the output functionals of interest.

Figure 5 shows the histogram of refinement parameters in a mesh which has achieved equidistribution. Since all cells in the domain have the same error, they fall in the same bin, and the histogram looks like a delta function whose height is N_{cells}.

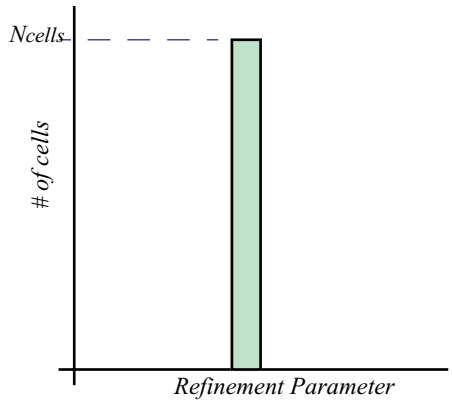


Figure 5: Idealized histogram of refinement parameters in a mesh which has achieved equidistribution.

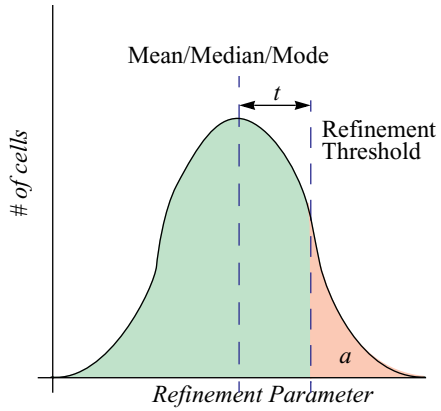


Figure 6: Idealized distribution of refinement parameters prior to *h*-refinement. Region *a* contains N_a cells.

Like most strategies for adaptation, the paradigm of Alg. **A** is to start with some initial distribution of refinement parameters, and drive this distribution toward the idealized distribution shown in fig. 5.

Figure 6 shows the Gaussian-like distribution of refinement parameters which serves as a model for the histogram prior to *h*-refinement. A common approach found in the literature is to set the refinement threshold to some fraction of a standard deviation above the mean of the distribution.[3][4][26]

3.2 A Fresh Look at Refinement Histograms

Figure 7 shows an actual refinement histogram resulting from a coarse grid simulation (3775 cells) of flow over an ONERA M6 wing at transonic conditions[25]. This plot bears little resemblance to the idealized Gaussian-like model shown in figure 6. After normalization, the refinement parameters lie between 0 and 1. The mean value is 0.011 but the standard deviation 0.04. Moreover, fully 82% of the cells lie below the mean, and almost 50% have $|\tilde{r}| < 0.001$. As a consequence of this extreme disparity in scales, setting the threshold, *t*, any place above the mean addresses the error in only a handful of the most severe cells. Only after the very worst errors are reduced by many cell refinements will error in the bulk of the domain be addressed. In shocked flows, the refinement parameter will be highest in cells with shocks or other strong

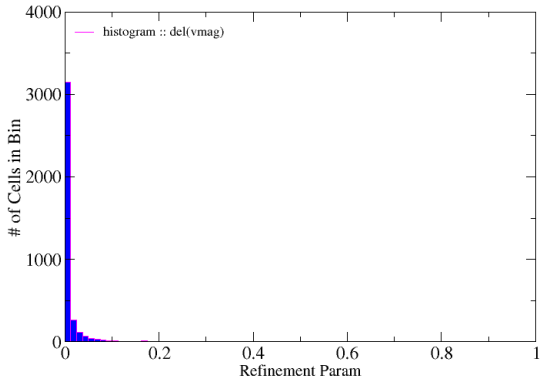


Figure 7: Histogram of adaptation parameter for a coarse mesh simulation of flow around an ONERA M6 wing at transonic conditions[25].

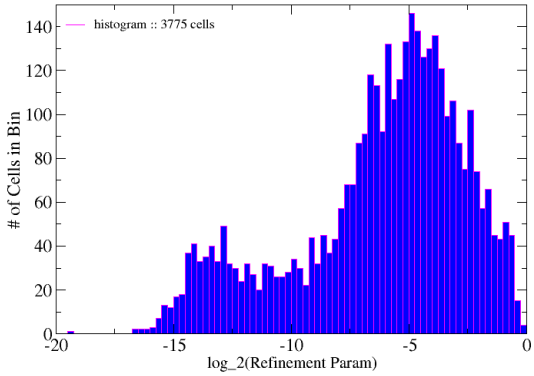


Figure 8: Histogram of data from fig.7 computed using $\log_2(|\hat{r}|)$ rather than the raw refinement parameter.

non-linear features. As a consequence, this approach will inadvertently result in a refinement process which over-resolves shocks and other severe features without ever addressing smooth regions of the flow. Oversights such as these have been shown to produce an adaptive procedure which can actually converge to the wrong solution[26].

In earlier work[4] we advocated a filtering approach which removed cells containing shocks or other strong features in an attempt to clean up the histogram prior to computing the mean and standard deviation. Even this approach, however is dubious given the huge disparity in scales.

An alternative approach for compressing the scales in the distribution is to simply take the log of the refinement parameter prior to binning up the histogram. Figure 8 shows the histogram of the same data as in figure *7* but computed using $\log_2(|\hat{r}|)$ rather than simply $|\hat{r}|$. The mean value of this new distribution is -6.4 and the standard deviation is 4.3. The rescaled data much more closely resembles the idealized distribution in figure 5 and values of the mean, median and mode are within 1.5 units of each other.

3.3 Optimal Threshold Selection

The motivation for choosing base 2 for the logarithms used to rescale the refinement parameters in the proceeding section becomes clear when selecting an appropriate threshold. Near grid convergence, each 2:1 cell refinement using a p^{th}-order scheme will reduce the *LTE* by a factor of 2^p. With 2:1 cell refinement and the present second-order scheme, the children of an *h*-refined cell will therefore get translated an absolute distance of 2 units to the left on these base-2 histograms. Figure 9 illustrates this process. If there are N_a cells in region *a* of figure *9*, and each cell is subdivided into *m* children, then *a** will contain $m\ N_a$ new cells.

If our goal is equidistribution, then we desire to build the delta function of figure 5. An optimal method constructs these as rapidly as possible. Assuming that the histogram is decreasing to the right of the modal value, the new histogram grows most rapidly if the highest point of *a** is placed on top of the mode of the existing distribution. Since the cells in *a* move 2 units to the left, the threshold which builds the highest new peak is identically 2.

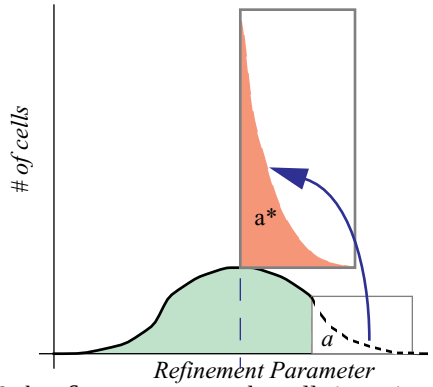


Figure 9: h-refinement moves the cells in region *a* to the left according to the order of accuracy of the scheme. If *a* contains N_a cells, then *a** contains $m\ N_a$ cells, where *m* is the number of children produced by refining a cell.

Subsequent refinements will add to this same peak, and the target level of error will remain constant as the mesh evolves. It is interesting to note that if the threshold is chosen above or below than this value the target error level will continue to migrate higher or lower (respectively) with subsequent cell refinements.

Just as refinement moves cells to the left on the histogram, coarsening transfers cells to the right. In the absence of coarsening, these low-error cells will remain in the histogram and appear as a “tail” to the left of the peak value. Figure 10 illustrates the evolving histogram. With the threshold chosen as described above, newly refined cells will not alter the histogram to the left of the peak value, and therefore no newly refined cells can ever end up in this tail. Since they were cells in the original unadapted mesh, this tail contains only coarse cells, and since coarse cells fill space very quickly, there cannot be very many of them in the domain as compared to the number of highly refined cells in the peak. In addition, since the peak was built via cell subdivision, and the number of children produced per cell is generally constant, the growth approaching the peak from the left will be very raipd.

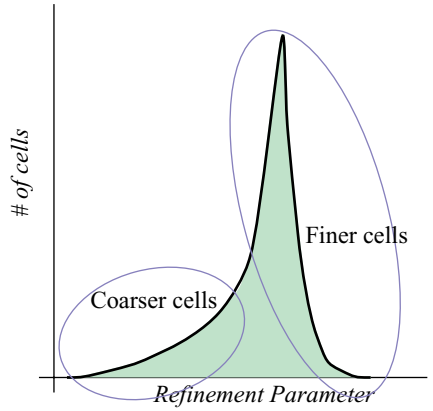


Figure 10: Evolution of a histogram for a mesh without coarsening.

Skeptics may point out that setting the adaptation threshold to its optimal value removes the user's "control" of the adaptive process. While this is precisely the goal of automation, there is a clear need for a user to be able to have some control over the level of error in the final solution.

Since the location of the new peak can be controlled by adjusting the threshold, *t*, the most efficient way to establish a desired location of this peak is to set it as early as possible (*i.e.* the first adaptation cycle). Subsequent adaptations will then continue to build on this same peak using the optimal threshold. This allows the user to set a desirable error level based upon the histogram of the unadapted coarse mesh, and then drive the refinement hands-free.

3.4 An Illustrative Example

Using the coarse mesh simulation from the base-2 histogram in figure 8, figure 11 shows the evolution of the histogram in this simulation over the next 5 adaptation cycles. This example clearly shows the rapid growth of the peak in the histogram confirming its approach toward equidistribution of the refinement parameter.

Since this is a real 3-D transonic flow, several issues merit discussion. Adaptation was driven by the undivided first difference of velocity magnitude scaled by the local mesh dimension as presented in §2.6. The adaptive procedure in Algorithm **A** tagged cells according to adaptation criteria, and these tags were then modified to satisfy the smoothness and mesh validity rules detailed in **A.2**. The adaptation criteria used in this example was:

$$a(\hat{r}_j) = \left\{ \begin{array}{ll} 1 & |\hat{r}_j| > \overline{|\hat{r}|} + 2 \\ 0 & \text{otherwise} \end{array} \right\}, \qquad (11)$$

where $\overline{|\hat{r}|}$ is the mean value of the magnitude of the refinement parameter rather than the precise modal value as called for by the theoretical development earlier in this section. Of course for the narrow peaked histograms shown in the figure, the mean is close to the mode - it is within one unit at every cycle after the first. Nevertheless one would expect somewhat better performance if the location of the true peak was used.

4 Numerical Examples and Discussion

The *LTE* estimates and feature detection approaches in §2 have been applied to both simple and complex configurations in three dimensions. This section begins by presenting results showing that, when combined with the *h*-refinement strategy of §3, both can provide valid approaches to achieve grid-converged solutions. We then present adapted solutions on two complex configurations to demonstrate the robustness of the procedure when run "hands-off" on real-world complex configurations.

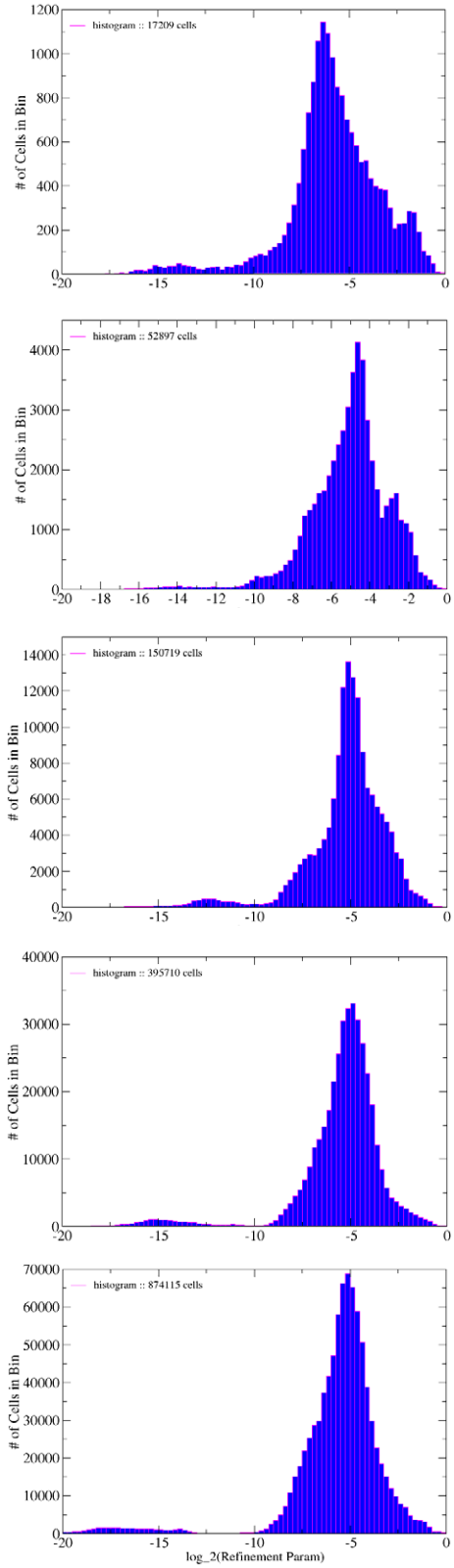


Figure 11: Evolution of histogram through 5 adaptation cycles for transonic ONERA M6 wing case using the optimal threshold.

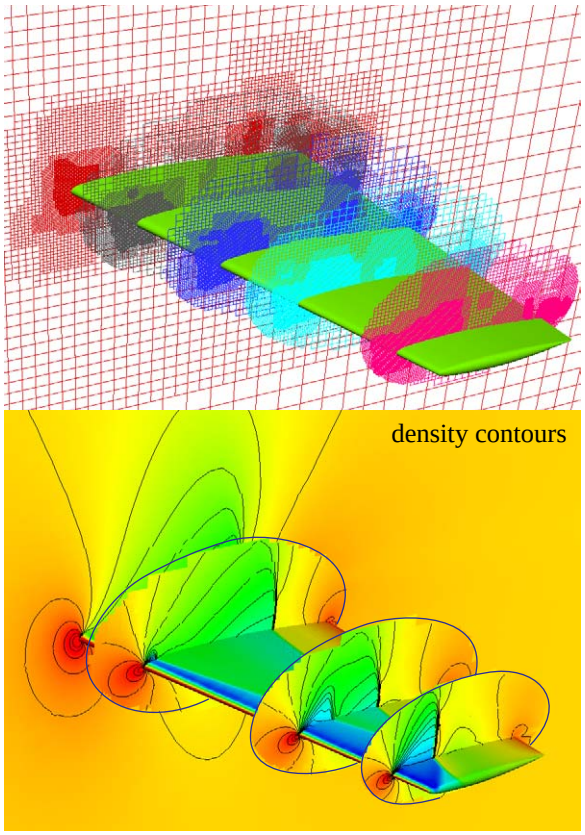


Figure 12a: Flow over an ONERA M6 wing at M_∞ =0.84 and α = 3.06°, adaptation driven by τ-estimates of LTE in density. The final mesh contains 1.8M cells

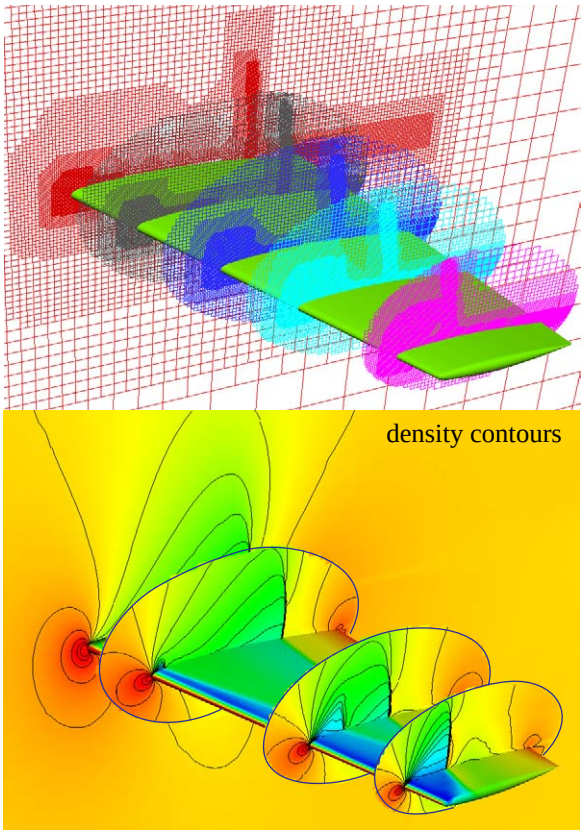


Figure 12b: Flow over an ONERA M6 wing at M_∞ =0.84 and α = 3.06°, adaptation driven using scaled first differences of velocity magnitude. The final mesh contains 1.9M cells

4.1 ONERA M6 Wing

In [17] the baseline Euler solver was validated using the well-known ONERA M6 wing example.[25] This case considers the transonic flow over this wing at M_∞ =0.84 and α = 3.06°. Although viscosity was obviously present in the experiment, the case has been widely studied using inviscid solvers, and a multitude of Euler solutions are available in the literature for comparison.

This flow was computed using both the multilevel τ-extrapolation *LTE* estimates and scaled first-difference (eq.10) based feature detection using $\phi = |\bar{V}|$ as a refinement parameter. Figure 12a displays both the mesh and solution resulting from the *LTE* based adaptation, while 12b contains these same plots using feature detection. In figure 12, the τ-extrapolation analysis used an initial mesh with 9 levels of geometry based refinement and then an additional two cycles of solution-based refinement following the philosophy of §2.4. The final mesh shown contains 1.8M cells. The feature detection based results in figure 12b began on a mesh created with 5 levels of geometry-based adaptation and about 30K cells followed by 6 cycles of adaptation. The final mesh contained 1.9M cells. After 5 levels of adaptive refinement, the mesh contained about 900K cells and integrated quantities (normal, axial and side force) were virtually the same as those obtained on the final (1.9M cell) mesh. The integrated quantities for both examples changed by less than 0.1% in the last adaptation cycle suggesting that the results are grid converged.

Lift and drag coefficients for τ-extrapolation were: 0.3041 and 0.0117 while those the feature-detection were 0.3042 and 0.0116. Comparison between the two simulations reveals a difference of less than 0.04% in the magnitude of the total force on the wing, and the final meshes have cell counts within 6% of each other.

Figure 13 shows convergence of the C_p profile at the 44% span station and includes an overplot of the experimental data

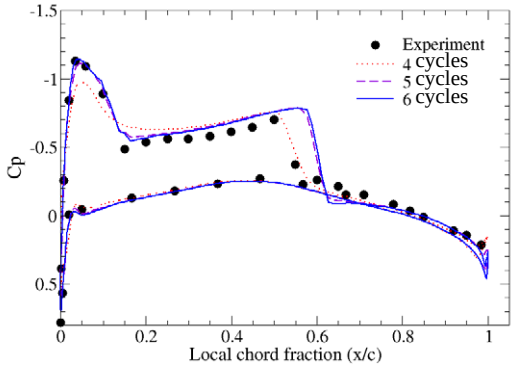


Figure 13: Pressure profiles from transonic ONERA M6 wing case at 44% span showing evolution of the C_p history over the last 3 adaptation cycles.

for the simulation in figure 12b[25]. Behavior at this span station is typical and shows convergence of the adaptive process over the last 4 adaptation cycles. In the figure, the profiles of the solution after 5 and 6 adaptation cycles are essentially indistinguishable. Comparison with the experimental data is generally good and shows the same discrepancies reported by other inviscid simulations of this viscous flow. In particular, the separation bubble following the rear shock is not modeled, and this shock is positioned slightly behind that in the experiment.

4.2 Complex Configurations

Figures 14 and 15 display the first of two examples showing real-world applications of the adaptation module on complex geometry. The supersonic canard-controlled missile geometry in figure 14a contains several features which make it challenging to simulate. At angle-of-attack, or any time the canards are deflected, they will create vortices which may interact with the tail fins of the missile. Due to the high fineness ratio of the missile, these vortices must convect over 30 canard chord lengths before reaching the tail. Clearly, excess numerical dissipation can easily destroy this important interaction. In addition, the disparity in length scales on the geometry makes this simulation challenging. The canard chord is only ~1/40th of the body length. The simulation must resolve not only fine geometric scales like the leading and trailing edges of the canards, but also the bow shock on the missile, and the shock system generated by the canards themselves. Inviscid overset (structured) grid simulations with the Army's OVERFLOW-D solver used over 30M points to resolve the features of this flow field.[27]

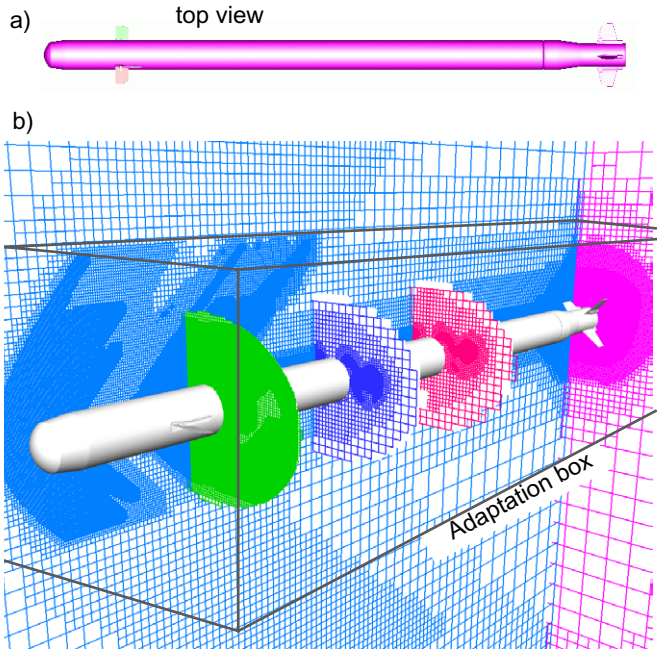


Figure 14: Geometry and adapted mesh for canard-controlled missile example. The final mesh has 4.5M cells and used 6 cycles of adaptive refinement, the last 3 of which were confined to the pre-specified adaptation box illustrated.

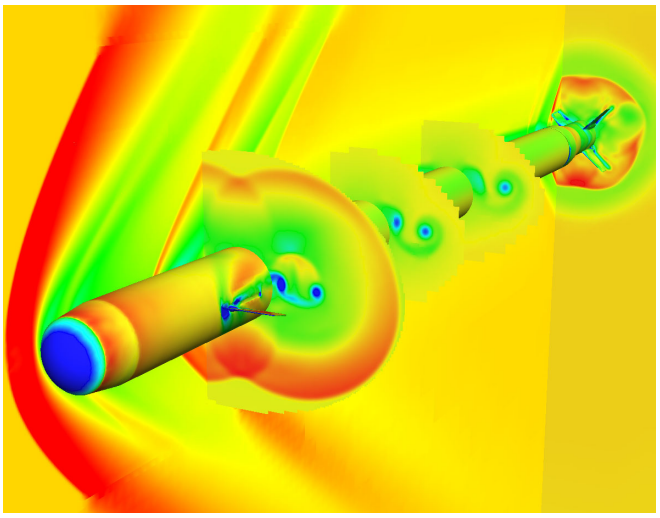


Figure 15: Velocity magnitude contours for flow over a canard-controlled missile of in fig. 14 at $M_\infty = 1.6$ at $\alpha = 3°$ with the canards deflected 15° (nose up).

Supersonic flow over this missile was computed at zero degrees roll, and $M_\infty = 1.6$ at $\alpha = 3°$. The canards are deflected 15° (nose up). These conditions give a reasonably strong interaction between the canard tip vortices and the leeward pair of the interdigitated tail fins.

Figure 15 shows contours of velocity magnitude in the discrete solution of this flow on the adapted mesh shown in figure 14. The refinement parameter in equation (10) based on density was used to drive the adaptive process. Both figures clearly display the trajectory of the canard vortex system as they convect down the body. The final mesh has 4.5M cells and used 6 cycles of adaptive refinement, the last 3 of which were confined to the pre-specified adaptation box illustrated. Several axial cutting planes in figure 15 display the evolution of the canard vortex as it travels down the missile body. The computed normal force coefficient on the final mesh matches the inviscid results in [27] to within 3%.

One interesting aspect of the missile simulation is that with the adaptive strategy outlined in §3 and the refinement parameter from equation (10), the canard vortex and other important smooth features in the flow are refined to the same level as the shocks. Thus the case can be made that the shocks are not receiving excessive attention from the refinement scheme.

This observation is further supported by the space-shuttle configuration displayed in figures 16-18. In this case the model was composed of 22 separate components, and included spoilers, flaps, rudders, engine bells, and other geometric detail. While the elevons and spoilers are nominally undeflected, some gaps exist, and there is flow leakage near these control surfaces.

The half-body, power-off configuration was simulated at $M_\infty = 1.5$ and $\alpha = 8°$, and refinement was focused in a box extending 3 body lengths in the crossfoot directions truncated just downstream of the orbiter. Figures 16-18 all display the

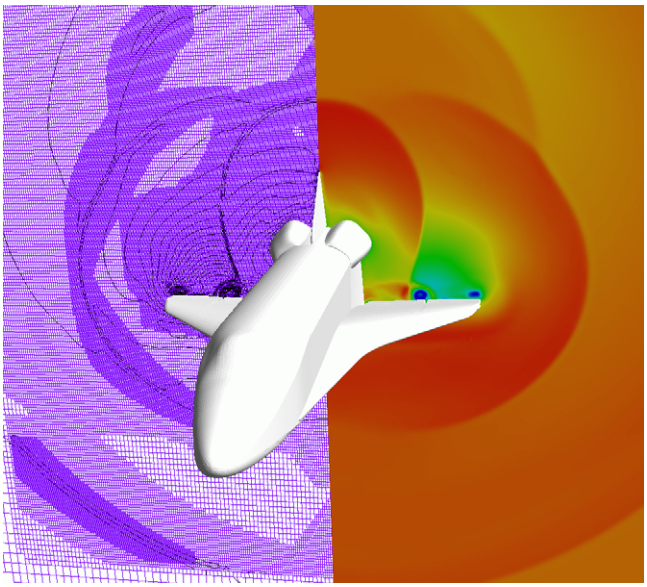


Figure 16: Computational mesh and velocity contours of solution for orbiter simulation at $M_\infty = 1.5$ and $\alpha = 8°$. The final mesh contains 8.5M cells.

solution using contours of velocity magnitude, and the scaled, undivided first difference of density was used as the adaptation parameter. Five cycles adaptation were carried out (hands-off) from an initially geometry refined mesh, producing a final mesh with 8.5M cells. Figure 16 shows a nose-on view of the grid mirrored to the starboard side, with contours of the solution on the port side. While the bow and wing shocks are reasonably well resolved, nearly equal mesh spacing is used in much of the near-body lee-side flow. The flow structure is reasonably complex, with the body, wing, OMS pods, etc. all producing massive curved shocks that are captured by the refinement. The curvature of these shocks results in strongly non-linear flow downstream and the refinement extends into these regions. In addition, both the wing tip vortex, and a vortex emanating from the gap between control surfaces are evident in this figure.

Figure 17 provides additional insight, displaying both the mesh and solution from a vantage point behind and above the port wing. The cutting plane in this view is located mid-way over the starboard wing, and some gaps between the control surfaces are visible. The bow, canopy, wing and trailing edge shocks are all clearly visible in this view. Figure 18 is a profile shot which contains a cutting plane through the symmetry plane to provides a better view of the canopy and bow shocks.

5 Conclusions and Future Work

We have presented Cartesian mesh adaptation strategies driven by either local truncation error estimates or feature detection. The adaptation module adds a solution-based mesh refinement capability to the geometry-based refinement of the Cartesian mesh generator. Both simple studies and highly complex 3D examples were presented with very high resolution, demonstrating the robustness and utility of the adaptation. The module produces several million Cartesian cells-per-minute on desktop computers and was demonstrated on complex example geometries with ~10^7 cells.

An interesting highlight of this work is an optimal strategy for *h*-refinement based on $\log_2(\,)$ histograms. This strategy avoids many of the pitfalls of the mean and standard devia-

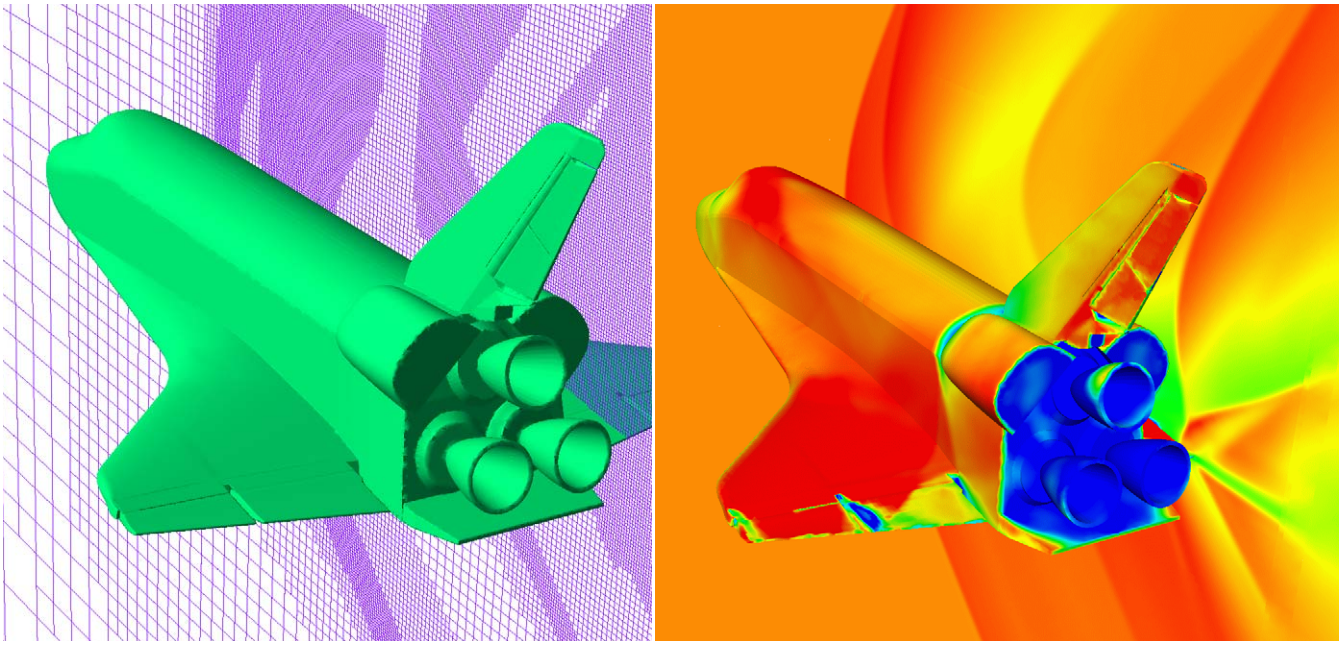


Figure 17: Rear three-quarter view of orbiter geometry and mesh showing gaps between control surfaces and cutting plane through solution at a mid-span location. $M_\infty = 1.5$, $\alpha = 8°$, velocity contours

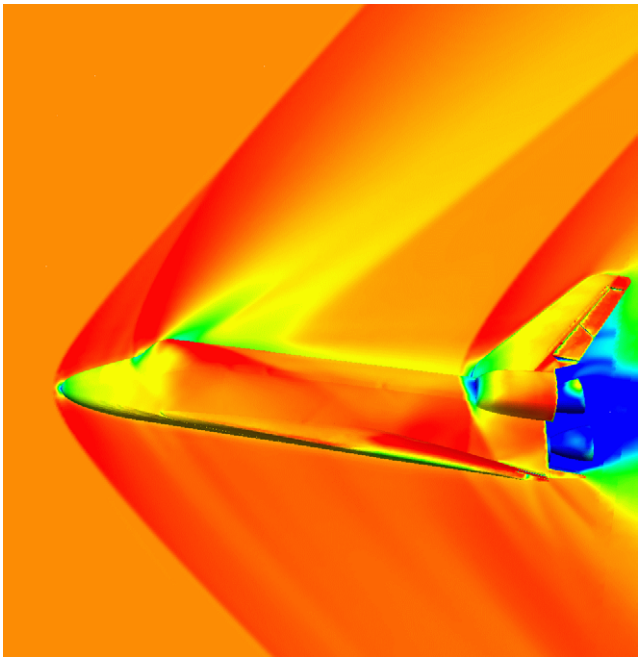


Figure 18: Side view of orbiter simulation and symmetry plane solution. $M_\infty = 1.5$, $\alpha = 8°$, velocity contours.

tion based approaches found in the literature. For hierarchal meshes, the approach is optimal in that it maximally equidistributes the refinement parameter for a given number of adaptation cycles. We believe it to be more reliable and robust than mean and standard deviation based approaches.

Our initial investigation of multilevel local truncation estimates was disappointing due to the irregularities in the mesh at the embedded boundaries and interfaces. While these complications can be overcome, they affect more of the mesh than was initially expected. More investigation of this problem is needed. When applied to a model problem with a known analytic solution, these truncation error estimates reconfirmed the accuracy advantages in both *LTE* and global error offered by Cartesian meshes.

Future work will focus on several outstanding topics. The behavior of the refinement strategy needs to be validated over a wider range of input conditions. While this strategy has been performed extremely well in initial investigations of flows with free stream Mach numbers from 0.8-1.6, it has not been exhaustively tested for broader conditions. The strategy assumes that the histogram of refinement parameters is monotonically decreasing to the right of the median bin, and the validity of this assumption should be investigated under more extreme conditions. A second topic for further investigation is selection of an initial threshold to control the overall error level in the simulation. Currently this is done by inspection of the coarse mesh histogram, but a more automated process would be desirable. We have also not implemented a mesh coarsening algorithm, and while this is not a major concern for steady flows, one will be needed for unsteady application in the future. Finally, adaptation is currently "focused" through the use of pre-specified adaptation regions. Such regions could be automatically generated using characteristic information from the flow to appropriately weight or unweight refinement parameters depending upon the cell's location in the domain and the input Mach number.

6 Acknowledgements

M. Berger was supported in part by AFOSR grant F49620-00-1-0099, and by DOE grants DE-FG02-88ER25053 and DE-FG02-92ER25139.

7 References

[1] Zienkiewicz, O.C., Zhu, J.Z., "A simple error estimator and adaptive procedure for practical engineering analysis." *Internat. J. for Num. Meth. in Eng.*, **24**:337-357, 1987.

[2] Dannenhoffer, J.F., and Baron, J.R., "Adaptation procedures for steady state solution of hyperbolic equations." *AIAA Paper -84-0005*, Jan., 1984.

[3] Kallinderis, Y.G., and Baron, J.R., "Adaptation methods for a new Navier-Stokes algorithm." *AIAA Jol.* **27,** Jan. 1985.

[4] Aftosmis, M.J., "Upwind method for simulation of viscous flow on adaptively refined meshes." *AIAA J.*, **32**(2):268-277. Feb., 1994.

[5] Nithiarasu, P., and Zienkiewicz, O.C., "Adaptive mesh generation for fluid mechanics problems." *Internat. J. for Num. Meth. in Eng.*, **47**:629-662, 2000.

[6] Aftosmis, M.J., Melton, J.E., and Berger, M.J., "Adaptation and surface modeling for Cartesian mesh methods." *AIAA Paper 95-1725CP*, Proc. of the 12th AIAA CFD Conf, San Diego CA. Jun., 1995.

[7] Peraire, J., Vahdati, M., Morgan, K., Zienkiewicz, O.C., "Adaptive remeshing for compressible flow computations." *J. Comp. Phys.*, **72**:449-466, 1987.

[8] Hecht, F., and Mohammadi, B., "Mesh adaptation by metric control for multi-scale phenomena and turbulence." *AIAA Paper 97-0859*, Jan., 1997.

[9] Fortin, M., Vallet, M.-G., Poirier, D., and Habashi, W.G., "Error estimation and directionally adaptive meshing." *AIAA Paper 94-2211*, Jun., 1994.

[10] Berger, M.J., and Oliger, J., "Adaptive mesh refinement for hyperbolic partial differential equations." *J. Comp. Phys.*, **33**:484-512, Mar., 1984.

[11] Berger, M., and Jameson, A., "An adaptive multigrid method for the Euler equations," *Lecture Notes in Physics 218*, Springer-Verlag. Proc. 9th Intl. Conf. Num. Meth. Fluid Dyn., June, 1984.

[12] Berger, M., and Jameson, A., "Automatic adaptive grid refinement for the Euler equations," *AIAA J.* **23**(4):561-568, April, 1985.

[13] Giles, M.B., "On adjoint equations for error analysis and optimal grid adaptation in CFD." in *Computing the Future II: Advances and Prospects in Computational*

Aerodynamics, eds. M. Hafez and D.A Caughey. John Wiley and Sons. 1998.

[14] Pierce, N.A., and Giles, M.B., "Adjoint recovery of superconvergent functionals from approximate solutions of partial differential equations." *Report No. 98/18*, Oxford Univ. Computing Lab. 1998.

[15] Venditti, D.A., and Darmofal, D.L., "A multilevel error estimation and grid adaptive strategy for improving the accuracy of integral outputs." *AIAA Paper 99-3292*. Jun., 1999.

[16] Venditti, D.A., and Darmofal, D.L., "Grid adaptation for functional outputs of 2-D compressible flow simulations." *AIAA Paper 2000-2244*. Jun., 2000.

[17] Aftosmis, M. J., Berger, M. J., and Adomovicius, G., "A parallel multilevel method for adaptively refined Cartesian grids with embedded boundaries." *AIAA Paper 2000-0808*, Jan. 2000.

[18] Aftosmis, M. J., Gaitonde, D., and Tavares, T. S., "Behavior of linear reconstruction techniques on unstructured meshes." *AIAA J.*, **33**(11):2038-2049, Nov. 1995

[19] LeVeque, R.J., "Numerical methods for conservation laws." *Lectures in Mathematics*, ETH Zurich. Ed. O. E Lanford. Birkhäuser Verlag, Boston. 1992.

[20] Ruppert, J., "A Delaunay refinement algorithm for quality 2-dimensional mesh generation." *J. Alorithms*, **18**(3):548-585, May 1995.

[21] Trottenberg, U., Schuller, A., and Oosterlee C., *Multigrid*, Academic Press, ISBN 012701070X, Dec., 2000.

[22] Propp, R.D., and Colella, P., "An adaptive mesh refinement algorithm for porous media flows", *LBL Technical Report LBNL-45143*, submitted to JCP.

[23] Wendroff, B., and White, A. B., "Some supraconvergent schemes for hyperbolic equations on irregular grids." *Second Intl. Conf. on Nonlinear Hyperbolic Problems*, Aachen, J. Ballmann and R. Jeltsch, eds., Vieweg, Braunschwieg, Germany, 1989.

[24] Morton, K.W., "On the analysis of finite volume methods for evolutionary problems", *SIAM J. Numer. Anal* **3**5(6), Dec. 1998.

[25] Schmitt, V., and Charpin, F., "Pressure distributions on the ONERA-M6-Wing at transonic Mach numbers." *Experimental Data Base for Computer Program Assessment*, AGARD Advisory Report AR-138, 1979.

[26] Warren, G.P., Anderson, W.K, Thomas, J.L., and Krist, S.L, "Grid convergence for adaptive methods." *AIAA Paper 91-1592-CP.*, June 1991.

[27] Nygaard, T. and Meakin, R., "Aerodynamic analysis of a spinning missile with dithering canards" *To Appear*, AIAA Applied Aero Conference, June, 2002.

8 Appendix

The supersonic-vortex model problem discussed in §2.1 and figures 2 and 3 was performed to measure the local truncation error of the embedded-boundary Cartesian solver. In this appendix, we present details of this investigation and compare the Cartesian results with 3 other meshing systems. In addition to the nested Cartesian grids, this experiment was performed using three popular body-fitted meshing schemes, and the same underlying solver (upwind, with linear-reconstruction). Regular (structured) quad, right triangular, and "quality" triangular[20] body-fit meshes were used. Figure A.1 shows the second-coarsest mesh of each of these types of grid. Four meshes of each type were used in the investigation and the meshes contained from 128 to 7809 control volumes. Special care was taken to match the numbers of control volumes for each mesh type as closely as possible.

The example was computed with an inner Mach number, $M_{in} = 3.0$, and taking $p_{in} = 1/\gamma$, $\rho_{in} = 1$, $r_i = 1$ and $r_o = 1.9$. Each case was initialized with the exact solution and the *LTE* (of density) within each cell was computed using eq.(4) by applying the residual operator without using flux limiters.

Table A.1 contains the L_1 norm of the *LTE* for each of the 16 simulations. The simulations for each of the mesh types correlated closely to a straight line, and the asymptotic slopes of each are given in the table.

Some aspects of these data merit discussion. As noted in §2.2, the rate of convergence of the *LTE* for all mesh types are similar except for that of the "quality" triangular mesh. While all the other mesh types demonstrated second-order accuracy, results for this grid system were only slightly better than first-order. Since this mesh is not quite uniform, the stencils used for the gradient estimation and reconstruction

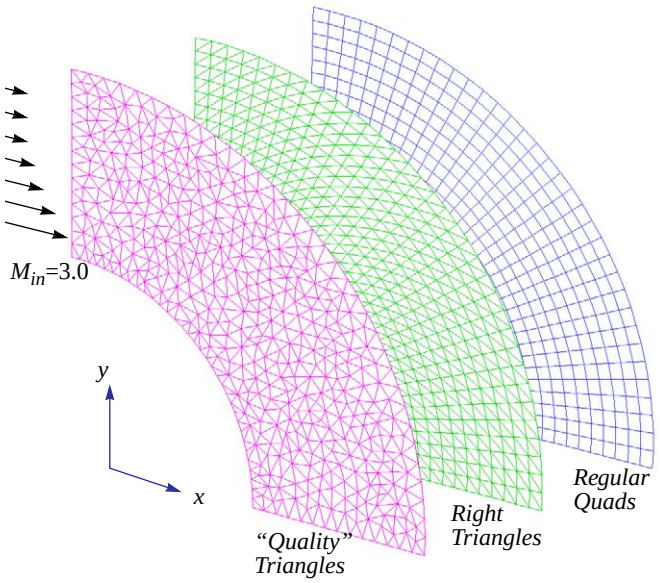


Figure A.1: Representative "Quality" Triangular, Right Triangular, and Regular Quad body-fit meshes used in *LTE* investigation. These meshes are the second coarsest used and have 505, 525 and 525 control volumes (respectively).

on all of the "quality" meshes are all slightly irregular. Since every stencil is irregular, each is polluted to some degree by stretching, cancellation of errors cannot occur, and the result is a marked degradation in accuracy.

These results contrast with earlier results for a similar problem using nearly-equilateral triangles[18]. That investigation showed that regular equilateral triangles performed as well (or better) than regular quads or right triangles. In this case, however, the "quality" mesh is not equilateral, although all the triangles are well formed as guaranteed by the Ruppert's 2-D delaunay technique in ref.[20]. Quality 2-D meshes were chosen for this investigation since it is not possible to generate uniform meshes of equilateral tetrahedra in 3-D. As a result, tetrahedral mesh generators typically resort to producing "quality" meshes that guarantee some angle criterion is everywhere met, just as Ruppert's Delaunay algorithm does in 2-D.

The structured quad and right-triangular meshes are substantially smoother than the quality triangular meshes. Nevertheless the *LTE* measurements indicate that even the mild irregularity in their stencil degrades their performance. While both provide second-order accuracy, the absolute error level is from 6 to 10 times worse than the Cartesian grid's performance where irregularity is confined to the boundary.

Cartesian Mesh with Embedded Boundary

# of Control volumes	*Measured L_1 (density) Error*	
138	0.03065	
507	0.00930	
1928	0.00246	
7549	0.00059	Asymptotic slope = 2.11

Body-Fit Right Triangular Mesh

# of Control volumes	*Measured L_1 (density) Error*	
144	0.37926	
525	0.07571	
2001	0.01565	
7809	0.00347	Asymptotic slope = 2.28

Body-Fit Structured (Quad) Mesh

# of Control volumes	*Measured L1 (density) Error*	
144	0.30998	
525	0.09223	
2001	0.02422	
7809	0.00629	Asymptotic slope = 1.94

Body-Fit Quality Triangular Mesh

# of Control volumes	*Measured L_1 (density) Error*	
128	0.52552	
505	0.22529	
1918	0.11936	
7490	0.05940	Asymptotic slope = 1.02

Table A.1: L_1–Norm of *LTE* in density for each of the 16 meshes used in the supersonic vortex investigation. The "Quality" triangulation meshes were produced using the quality Delaunay triangulation algorithm of ref.[20] and had no angle less than 29°.